Robot Behavior-Based User Authentication for Motion-Controlled Robotic Systems

Long Huang*, Zhen Meng†, Zeyu Deng*, Chen Wang*, Liying Li‡, Guodong Zhao†
*School of Electrical Engineering and Computer Science, Louisiana State University, Baton Rouge, LA 70820 USA
Email: lhuan45@lsu.edu, zdeng6@lsu.edu, chenwang1@lsu.edu
†James Watt School of Engineering, University of Glasgow, Glasgow, G12 8QQ Scotland
Email: 2227311M@student.gla.ac.uk, Guodong.Zhao@glasgow.ac.uk
‡Mathematics, Physics and Electrical Engineering, Northumbria University, Newcastle, Newcastle upon Tyne, United Kingdom
Email: emma.li@northumbria.ac.uk

***Abstract*—Motion-controlled robotic systems would become more and more popular in the future since they allow humans to easily control robot to carry out various tasks. However, current authentication methods rely on static credentials, such as password, fingerprints, and faces, which are independent of the robot control. Thus, they cannot guarantee that a robotic is always under the control of its enrolled user. In this paper, we build a motion-controlled robotic arm system and show that a robotic arm's motion inherits much of its user's behavioral information in interactive control scenarios. Based on that, we propose a novel user authentication approach to verify the robotic arm user. In particular, we log the angle readings of the robotic arm's joints to reconstruct the 3D movement trajectory of its end effector. We then develop a learning-based algorithm to identify the user. Extensive experiments show that our system achieves 95% accuracy to verify users while preventing various impersonation attacks.**

***Index Terms*—interactive control, robot behavior, network controlled robot, user authentication, cyber-physical security**

I. Introduction

Motion-controlled robotic systems, in particular the robotic arm systems, have been increasingly used for a multitude of applications for providing augmented interactions [1], including education, research, industrial control and social network. A typical motion-controlled robotic arm system comprises two end devices connected by a network. The client-end device tracks the user's motions and issues the corresponding control commands. The robotic arm at the other end receives and executes the commands. In the meanwhile, the user observes the robotic arm's movements and adjust his/her hand motions accordingly for interactive control, which facilitates performing fine tasks.

To make sure the robot is under the control of enrolled user(s), existing authentication approaches rely on passwords or physiological biometrics (e.g., fingerprints and faces) for system logins [2]. However, these authentication methods based on static credentials are independent of the robot control. They could hardly guarantee that the robotic arm is truly under the control of the enrolled user(s). For example, an adversary may substitute the user's control commands with malicious ones to bypass the above authentications. Moreover, the static authentication inputs could be obtained or forged by an adversary to fool the system.

The behavioral biometric authentication verifies a user via human body motions. In particular, the behavioral characteristics of the user's 3D hand gestures can be captured and verified by vision sensors [3], wearable inertial sensors [4] and radio signal sensors [5]. However, these methods are based on the user's motion data captured at the user end. They are hard to protect the robotic arm at the other end of the network. More specifically, it is still hard to know whether the robot control commands really come from its enrolled user and are not altered before they reach the robot.

To the best of our knowledge, this is the first study on user authentication for a motion-controlled robotic arm system, where the robotic arm's motion behaviors are leveraged to verify the user. In this study, we first build up a real motion-controlled robotic arm platform, consisting of a 7-Degree-Of-Freedom (DOF) robotic arm at the robot end, six OptiTrack motion capture sensors at the user end, and a local area network connecting the two ends. In particular, we implement a human-robot Kinematic mapping algorithm to enable real-time robot control. Our experiment shows that the robotic arm inherits its user's unique behavioral characteristics in the interactive control environment.

Based on this, we propose a novel user authentication approach that embeds the robot control into user authentication, which significantly enhances the system security. Specifically, our approach first exploits the joint angle readings read from the robotic arm to reconstruct its end-effector trajectories. Then, we derive the unique features to capture the user's behavioral characteristics carried by the robotic arm. Moreover, a machine learning algorithm is designed based on weighted Dynamic Time Warping (DTW) to verify the identity of the human controller (i.e., user). Our experiment results demonstrate the promising authentication performance.

Our contributions are summarized as follows:

- We propose a robot behavior-based user authentication approach to embed robot control into user authentication, which can effectively protect the access to motion-controlled robotic arm systems.

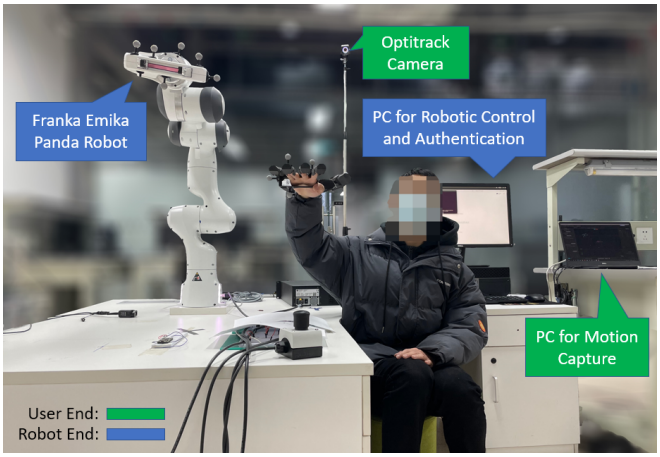

Fig. 1: The motion-controlled robotic arm platform.

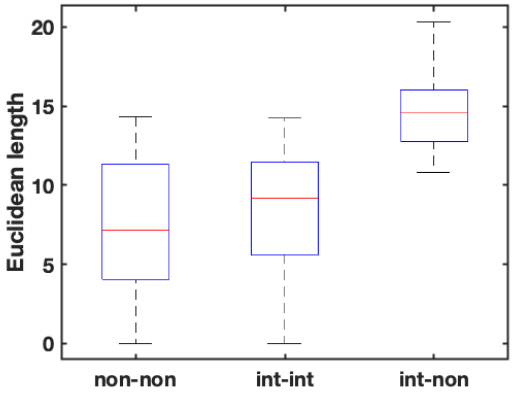

Fig. 2: Comparison of the user's motion behaviors under the *interactive control* (int) and the *non-control* (non) scenarios.

- We demonstrate that the robotic arm inherits its operator's behavioral biometric in the interactive control scenarios, whose behaviors can thus be used for authentication.
- We build up a real motion-controlled robotic arm platform to evaluate our authentication approach. The experiments with 10 participants in practical control scenarios show that our approach accurately verify users while preventing the various impersonation attacks.

II. System Description and Feasibility Study

In this section, we first introduce our motion-controlled robotic arm platform, which consists of user-end devices, robot-end devices, and a local area network. Then, we provide the feasibility study to show that the robotic arm inherits its user's unique behavioral characteristics in the interactive control environment. Finally, we introduce the brief system flow of the proposed user authentication approach.

A. Motion-Controlled Robotic Arm Platform

In this work, we develop a motion-controlled robotic arm platform to allow a user to interactively control a robotic arm in real time. As shown in Fig 1, the platform consists of the devices at two ends. The user-end devices capture the user's hand motions and generate the control commands based on the human-robot kinematic mapping [6]. The control commands are sent to the robot end via a local area network to control the robotic arm in real time. In the meanwhile, the user observes the robotic arm movements and adjust his/her hand motions to achieve the interactive control.

1) User-end Devices: At the user end, six OptiTrack cameras are deployed in a 4m by 4m square area to capture the user's hand movements. The user is required to wear a 3D-printed glove with 7 passive markers, which allows the OptiTrack system to obtain reliable measurements and construct a rigid body of the user's hand with less than 0.2 millimeter errors [7]. When the user moves the hand in the air, the 3D hand coordinate is obtained and sampled at 120 Hz. Then, the *personal computer* (PC) performs robotic arm inverse kinematics algorithm based on the Denavit–Hartenberg parameters of the robot arm, mapping each user's 3D hand coordinates to 7 joint angles [6].

2) Robotic Arm-end Devices: At the robotic arm-end, a Franka Emika Panda robot is used in the experiments [8]. A PC acts as the robotic arm controller for path planning. This is achieved by the *proportional–integral–derivative* (PID) control algorithm, converting the received joint angle values into a series of robotic control angular velocity commands within some trajectory limitations. Then, the angular velocity commands can be directly executed by the application programming interface (API) functions provided by libfranka, which is C++ implementation of *Franka Control Interface* (FCI) [9]. The interface allows the PC to update the target of each joint position and read the robot state to get the sensor data at 1 kHz. Since the updating rate of the angle data is 120 Hz, a new path planning is made at every 8 milliseconds.

3) Local Area Network: We use the commercial Ethernet to connect the two ends and facilitate the data transmission from the user-end PC to the robot-end PC. We utilize User Datagram Protocol (UDP) to support the high packet delivery rate. This protocol also reduces the queuing delay at the transmitter side and improves the real-time tracking performance. The system can also be easily deployed with the TCP/IP protocol. According to the specific application and performance requirements, other high-speed networks, such as LTE and 5G, can be used for our prototype.

B. Feasibility Study

1) Human Behaviors in Interactive Control Environments: Based on the above platform, we first investigate whether a user's motions exhibit behavioral uniqueness in interactive control environments and how it is correlated to the user's regular daily behaviors. In particular, we ask a participant to repeatedly draw an "S" in the air 40 times while controlling a robotic arm using the above platform. For comparison, the participant also draws the same "S" freely as he regularly does without operating a robotic arm. User's hand motions in both the interactive and non-control scenarios are captured by the

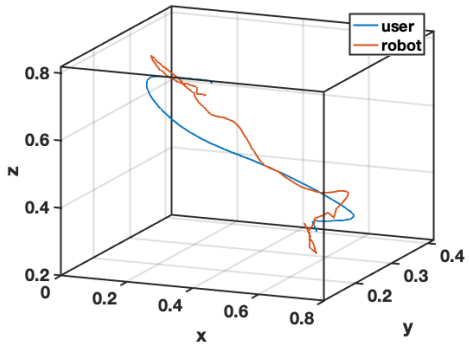

Fig. 3: Comparison of the user's hand and the robotic arm end-effector trajectories of writing an "s".

OptiTrack cameras and represented as the time series of 3D coordinates. We then compare the user's hand motions within each scenario and across the two scenarios by calculating their Euclidean distances. Fig. 2 presents the boxplots of the comparisons. We find that the inter-class Euclidean distances of *int-non* are much higher than that of the intra-class comparison *non-non*. This result shows that the user's motions under the interactive control scenario are much different from his regular behaviors. The reason is that the user has to adapt to the movements of the robotic arm during the interactive control, which changes the user's behavioral characteristics. Moreover, we find that the intra-class comparison *int-int* shows the low Euclidean distances comparable to *non-non*. This result indicates that the user's behaviors in the interactive control scenario are still consistent, which facilitates learning the user's interactive control behaviors for authentication.

2) Human Behavioral Biometrics Embedded in Robot Motions: We next investigate whether the robotic arm could inherit the user's behavioral biometrics. In particular, we compare the 3D coordinates of the human hand and the robotic arm end effector, which are captured by the OptiTrack cameras. As the robotic arm has more joints and different arm lengths and moves in a different coordinate, its motions could not be directly compared with human arm motions. Thus, we perform the interpolation, scaling and coordinate alignment (introduced in Section III-B) to normalize the two trajectories. The resulted trajectories are compared in Fig 3. We find that both trajectories show the curve of an "S" in the 3D space. Although the robotic arm follows the path of the user's hand, its movement trajectory does not completely replicate that of the user's hand. This is because the 7-joint robotic has a different kinematic mechanism compared to the human arm. Additionally, when following the user's hand motions, the sampling errors and the network traffic delays cause the robotic arm to show much more additional movements. But the robotic arm's trajectory still exhibits a certain similarity to that of the user's hand, because both the user and the robotic arm tries to adapt to each other's motion in the interactive control scenario. The result indicates that the robotic arm carries a portion of the user's behavioral biometrics, which can be used to verify its human operator and secure the robotic arm.

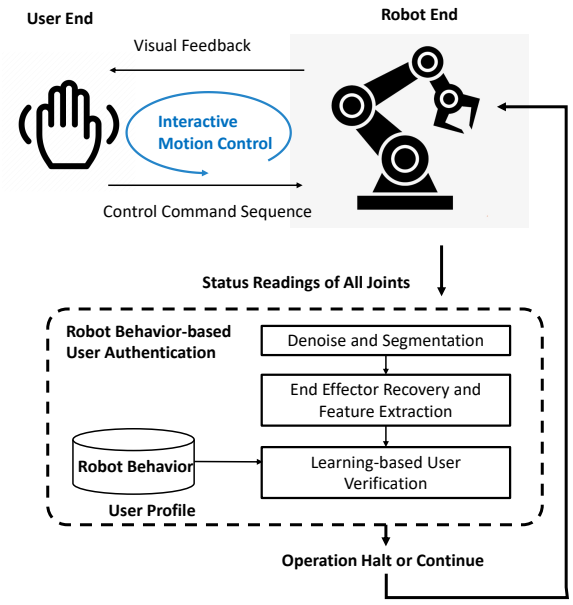

Fig. 4: The flow of the proposed user authentication for motion-controlled robotic arm.

C. System Flow

The basic idea of our approach is to verify the user based on the behaviors of the robotic arm under the interactive control. The system flow of the proposed robot behavior-based user authentication approach is shown in Figure 4, which is deployed at the robotic arm end. After logging into the platform from the user end using a password, the user can operate the robotic arm with hand motions to perform tasks in real time. During the control process, the control commands are generated and sent to the robot end. In the meanwhile, the user receives the visual feedback of the robotic arm's gesture for interactive control.

When the robotic arm executes the received control commands, the sampled angle values of its joints are taken as the input of our authentication approach. The joint readings are first segmented and normalized to represent each hand movement. Next, we reconstruct the moving trajectory of the robotic arm's end effector in the 3D space based on forward kinematics. We then derive unique features to capture the robotic arm's behavioral characteristics associated with the user's behavioral biometric, which are fed into a machine learning model. When enrolling the system and using the robotic arm for the first time, the robotic arm's behavior template is obtained to create the user's profile. When the user accesses the robotic arm at a later time, the current robotic arm behavior is compared with the user's profile to verify whether the user's identity is as claimed. Based on the verification result, the robotic arm would continue to operate or reject the access and halt while requiring a re-login.

III. User Authentication Design

A. Reconstructing the Robotic Arm's End-effector Trajectory

Our authentication approach utilizes the logged angle values of the robotic arm's seven joints to verify its user. To capture how the robotic arm follows the user's motion and inherits the user's behavior, we reconstruct the end-effector trajectory of the robotic arm based on forward kinematics. The robotic arm's motion mechanism is modeled as shown in Fig 5. For each joint such as Joint i, Z_i is its rotation axis, and X_i and Y_i are on its rotation plane. X_i is defined to be in the same direction of the arm link $\vec{l_i}$ (towards Joint $i+1$) at the beginning of the robotic arm control. When the robotic arm starts to move, the coordinate of Joint i can be determined by all of its $i-1$ prior joints. We then use the Denavit-Hartenberg [10] parameters to describe the robot kinematics. In particular, the link length $|\vec{l_i}|$ is the distance between Joint i and Joint $i+1$, while the link offset $\vec{d_i}$ is measured against the Z_i-axis. Figure 5 illustrates the joint status at time t, when the angle between Z_i and Z_{i+1} is α_i, and the arm link $\vec{l_i}$ rotates with the angle θ_i, The transformation matrix from Joint i to Joint $i+1$ can thus be expressed as:

$$
\begin{aligned}
{}^{i}T_{i+1} &= R_X(\alpha_i)D_X(|\vec{l_i}|)R_Z(\theta_i)Q_i(\vec{d_i}) \\
&= \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & \cos\alpha_i & -\sin\alpha_i & 0 \\ 0 & \sin\alpha_i & \cos\alpha_i & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & |\vec{l_i}| \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \\
&\begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & 0 \\ \sin\theta_i & \cos\theta_i & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & \vec{d_i} \\ 0 & 0 & 0 & 1 \end{bmatrix} = \\
&\begin{bmatrix} \cos\theta_i & -\sin\theta_i & 0 & |\vec{l_i}| \\ \sin\theta_i\cos\alpha_i & \cos\theta_i\cos\alpha_i & -\sin\alpha_i & -\vec{d_i}\sin\alpha_i \\ \sin\theta_i\sin\alpha_i & \cos\theta_i\sin\alpha_i & \cos\alpha_i & \vec{d_i}\cos\alpha_i \\ 0 & 0 & 0 & 1 \end{bmatrix},
\end{aligned} \tag{1}
$$

where R represents the rotation matrix, and D and Q denote the translation. The 3D coordinate of the end effector ${}^{n}pos$ is calculated based on the base point ${}^{0}pos = (x_0, y_0, z_0)^T$, which has a fixed location, as:

$$
{}^{n}pos = \begin{bmatrix} x_n \\ y_n \\ z_n \\ 1 \end{bmatrix} = {}^{0}T_n \begin{bmatrix} x_0 \\ y_0 \\ z_0 \\ 1 \end{bmatrix} = {}^{0}T_n \begin{bmatrix} 0 \\ 0 \\ 0 \\ 1 \end{bmatrix}, \tag{2}
$$

where the transformation matrix ${}^{0}T_n$ is obtained by

$$
{}^{0}T_n = {}^{0}T_1 \ {}^{1}T_2 \ {}^{2}T_3 ... {}^{n-1}T_n. \tag{3}
$$

The derived 3D coordinate time series $[{}^{n}pos_0, {}^{n}pos_1, ..., {}^{n}pos_t]$ describe the movements of the robotic arm, based on which we further extract the robotic arm's unique behaviors.

B. Data Calibration and Normalization

Due to user's varying hand movement speeds and scales, orientations and distances to the motion capture device, the derived robotic arm movement trajectories must be normalized and calibrated before we distinguish the minute behavioral differences among users. To address the above issues, we apply re-scaling and axes alignment to the reconstructed end-effector trajectory. In particular, the end-effector trajectory of each movement is first segmented based on the short-time energy. We then perform the interpolation to unify all end-effector trajectories into the same length in time. Furthermore, The 3D space that encloses a trajectory is scaled to a $1 \times 1 \times 1$ bounding box. Additionally, to unify the orientations of different trajectories, we pre-define a reference direction and rotate all the trajectories to align with it.

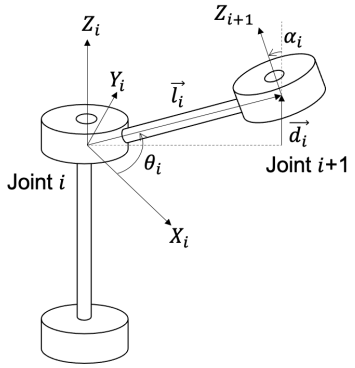

Fig. 5: Kinematic model of the robotic arm.

C. User Authentication

As the user could operate the robotic arm to perform different tasks, we propose to exclude the impacts of the task differences to focus better on the robot's behaviors. Thus, we first recognize which task the robotic arm is performing and then utilize the user's template of the identified task to verify the user. To achieve the goal, we develop a weighted DTW-based method to verify the user in two steps and derive unique features to capture both the task differences and the individual behaviors embedded on the robotic arm.

Feature Extraction. We derive six types of feature sequences from the end-effector trajectory of the robotic arm, including 3D coordinates, coordinate differences, velocities, accelerations, slope angles and curvatures. The resulted eleven feature sequences f_k, $k = 1, ..., 11$ are used to train the robotic arm task classification model and the user verification model respectively.

Robotic Arm Task Recognition. During the training phase, we construct the task templates $f_{k,task}$ based on a number of users' various task instances. Moreover, we assign weights $w_{k,task}$ to the different task feature sequences according to their independent performance [11] to distinguish the robotic arm tasks. Given a robotic arm end-effector trajectory in the testing phase, we compute its weighted DTW distance to each of the task templates as

$$
\sum_{k=1}^{11} DTW(f_{k,test}, f_{k,task}) \times w_{k,task}. \tag{4}
$$

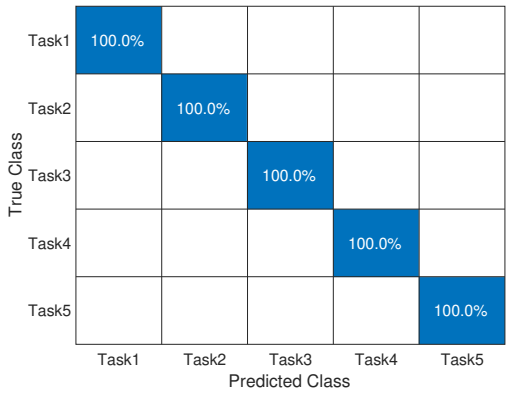

Fig. 6: Confusion matrix of task classification.

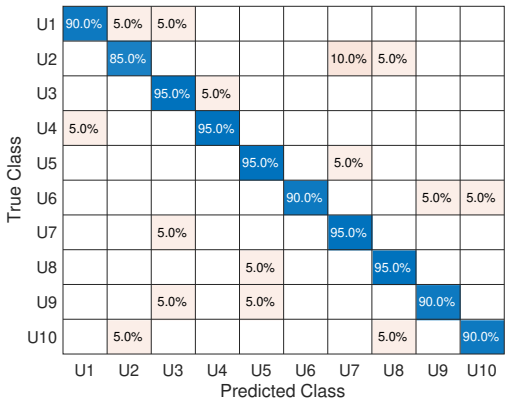

Fig. 7: Performance of user classification.

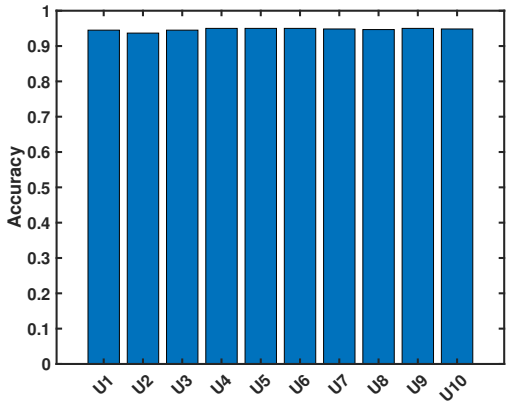

Fig. 8: Performance of user verification.

The robotic task is recognized to be the class, which exhibits the minimum weighted DTW distance.

User Verification. After the task is identified, we utilize the user's template of the task to verify the user's identity. The user's template of a task is obtained based on the user's multiple instances of performing the same task. The same types of features are used but the weights $w_{k,user}$ are calculated based on the performance of each feature sequence to distinguish the users under the task. During the user verification, we compute the weighted DTW distance using the user's template as

$$\sum_{k=1}^{11} DTW(f_{k,test}, f_{k,user}) \times w_{k,user}. \quad (5)$$

If the result is less than a threshold, the robotic arm is believed to be under the control of the legitimate user. Otherwise, the verification fails, the robotic arm operation is aborted.

IV. Performance Evaluation

A. Experimental Setup

To evaluate our authentication approach, we recruited 10 participants for the experiment, who were all first-time users of our platform introduced in Section II-A. They were given several minutes to get familiar with the platform before data collection. We then asked them to perform five basic tasks by operating the robotic arm, including moving the robotic arm along straight lines and snake-like, wavy, zigzag and circular curves (i.e., "–", "S", "W", "Z" and "O"). Each type of task was performed 40 times by each participant. The data collection lasted for five months. The resulted data set is split by half for training and testing respectively.

B. Robotic Arm Task Classification

We first present the performance of our approach to recognize different robotic arm tasks. The confusion matrix in Figure 6 shows that our approach distinguishes five basic tasks with 100% accuracy for all the participants. The result indicates that the logged joint states precisely describe the robotic arm's motions when the user performs different tasks, which enables us to further analyze the minute behavioral differences of a robotic arm under each specific task.

C. User Authentication

1) Classification: We first present the performance of our approach to distinguish users, which reflects how the robot behaviors are distinct among users. The confusion matrix of user classification based on the robot behaviors of all five tasks is as shown in Figure 7. We observe that nine of the ten users are classified with over 90% accuracy, and five users achieve 95% accuracies. The median classification accuracy is 92.5% The results show that our approach accurately distinguish the users based on the robot behavior.

2) Verification: In the practical scenario, the user must log in the platform with credentials before using the robotic arm. Thus, the user authentication becomes to verify whether the user's identity is as claimed by the login. We present the verification accuracy for each user regarding all five types of tasks in Figure 8. We observe that the verification accuracies for all the participants are between 93.7% and 95%. The median verification accuracy for each user is 94.8%. The result confirms that our approach could accurately verify the robotic arm user based on the robot behavior.

D. Impact of Training Data Size

We next study the impact of the training data size on our approach performance. Figure 9 shows the user verification performances when 5 to 20 instances are utilized for training

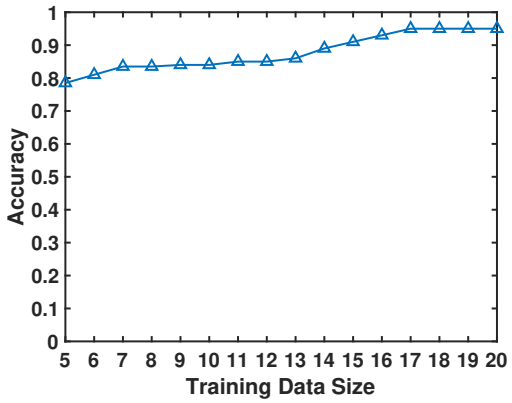

Fig. 9: Performance under different training size (verification).

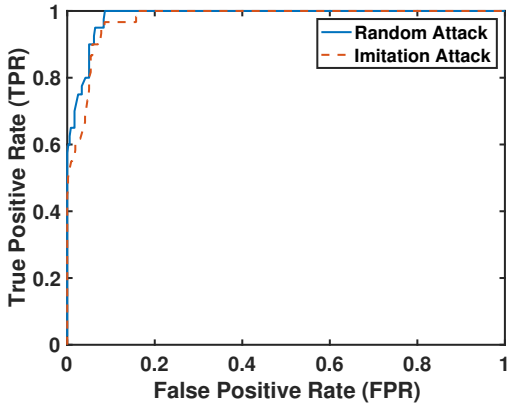

Fig. 10: ROC curves of defending attacks.

respectively. We observe that our approach achieves a good performance with a small training data set. In particular. Our approach achieves over 80% accuracy when the machine learning model is trained with 5 or more task instances. The user verification accuracy increases with the enlarged training data size. When the 7 and 15 task instances are used for training, the user verification performances are 84% and 91% respectively. The accuracy reaches 95% when 17 task instances are used. The results indicate that our approach is capable of verifying the user without requiring much training effort.

E. Performance Under Attacks

At last, we evaluate our approach under two types of attacks. We assume the attackers have gained the access to the robotic arm platform with correct logins to operate the robotic arm. Then, the adversary could use his/her own behavior to control the robotic arm for *random attack*. A skilled adversary who have the knowledge of the user's behavior could further imitate the user's behavior while operating the robotic arm for *imitation attack*. We asked six participants to act as the attackers for both attacking scenarios, and each attacker attempts to impersonate the other 9 participants. To perform the imitation attack, the attackers learn the legitimate users' behaviors from their robotic arm control videos.

The ROC curves of our approach to defend the two types of attacks are shown in Figure 10. We find that our system achieves high performances to prevent the two types of attacks. In particular, our system achieves a 95% True Positive Rate (TPR) to verify the legitimate users under random attack, while the False Positive Rate (FPR) is 6%. The Equal Error Rate (EER) is 6.3%. Under the imitation attack, our approach achieves a 97% TPR and a 9% FPR. The EER is 8.9%. The results indicate that it is hard for an adversary to imitate the user's interactive control behaviors, and the robot behaviors can be used to effectively prevent impersonation attacks.

V. Conclusion

In this work, we demonstrated that the human behaviors in the interactive robot control environment are distinguishable, and the robotic arm inherits much of the user's behaviors, which can be leveraged for authentication. We proposed a novel user authentication approach for the motion-controlled robotic arm by examining the robotic arm's joint status. In particular, we reconstructed the robotic arm's end-effector trajectories from its joint angle values, and derived features to capture the robot's unique behaviors corresponding to its user's motions. Based on that, we designed a weighted DTW-based algorithm, which first recognizes the robotic arm task and then verifies the user who is controlling the robotic arm. We built a real motion-controlled robotic arm platform to evaluate our approach. Experiments showed that our authentication approach verifies the user with 95% accuracy based on the robot behavior.

References

[1] J. Rekimoto and K. Nagao, "The world through the computer: Computer augmented interaction with real world environments," in *Proceedings of the 8th annual ACM symposium on User interface and software technology*, 1995, pp. 29–36.

[2] V. M. Vilches, L. A. Kirschgens, A. B. Calvo, A. H. Cordero, R. I. Pisón, D. M. Vilches, A. M. Rosas, G. O. Mendia, L. U. S. Juan, I. Z. Ugarte *et al.*, "Introducing the robot security framework (rsf), a standardized methodology to perform security assessments in robotics," *arXiv preprint arXiv:1806.04042*, 2018.

[3] J. Tian, C. Qu, W. Xu, and S. Wang, "Kinwrite: Handwriting-based authentication using kinect." in *NDSS*, vol. 93, 2013, p. 94.

[4] D. Lu and D. Huang, "Fmcode: A 3d in-the-air finger motion based user login framework for gesture interface," *arXiv preprint arXiv:1808.00130*, 2018.

[5] H. Abdelnasser, M. Youssef, and K. A. Harras, "Wigest: A ubiquitous wifi-based gesture recognition system," in *2015 IEEE Conference on Computer Communications (INFOCOM)*. IEEE, 2015, pp. 1472–1480.

[6] L. Sciavicco and B. Siciliano, "Modelling and control of robot manipulators," *Industrial Robot*, vol. 11, no. 12, p. 1828, 2000.

[7] I. D. O. NaturalPoint, "Optitrack prime 13," Last visited: May 2020, https://www.optitrack.com/cameras/primex-13.html.

[8] F. E. GmbH. Franka Control Interface Documentation. (Dec 2020). [Online]. Available: https://frankaemika.github.io/docs/overview.html

[9] F. E. GmbH. Requirements on communication interface for panda research robot. (Dec 2020). [Online]. Available: https://frankaemika.github.io/docs/requirements.html

[10] J. Denavit and R. S. Hartenberg, "A kinematic notation for lower-pair mechanisms based on matrices," *Trans. ASME E, Journal of Applied Mechanics*, vol. 22, pp. 215–221, June 1955.

[11] S. Celebi, A. S. Aydin, T. T. Temiz, and T. Arici, "Gesture recognition using skeleton data with weighted dynamic time warping." in *VISAPP (1)*, 2013, pp. 620–625.